**Name: Ali Hassaan Mughal**

**Registration: 173627**

**CS6B**

* **Is it better to use multithreaded version compared to single threaded in PI calculation?**

**No, the multithreaded version in this case takes more time.**

**Task 1:**

**Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Diagnostics;

namespace ConsoleApplication1

{

class Program

{

static long num\_Steps = 10000000;

static double steps;

static double[] values;

static void Main(string[] args)

{

Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();

int i = 0;

double x, pi, sum = 0.0;

steps = 1 / (double)num\_Steps;

for (i = 0; i < num\_Steps; i++)

{

x = (i + 0.5) \* steps;

sum = sum + 4.0 / (1 + x \* x);

}

pi = steps \* sum;

Console.WriteLine("Pi Simple = ", pi);

stopWatch.Stop();

TimeSpan ts = stopWatch.Elapsed;

string elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",

ts.Hours, ts.Minutes, ts.Seconds,

ts.Milliseconds / 10);

Console.WriteLine("RunTime " + elapsedTime);

Console.Write("Enter a key to end");

Console.ReadKey();

}

}

}

**Output:**



**Task 2:**

**Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Diagnostics;

namespace ConsoleApplication1

{

public class ThreadPi

{

private double x, steps, sum = 0,pi;

private int start\_i, end\_i;

public ThreadPi(int start\_i, int end\_i, double steps)

{

this.start\_i = start\_i;

this.end\_i = end\_i;

this.steps = steps;

}

public void ThreadFunction()

{

for (int i=start\_i;i<end\_i;i++)

{

x = (i + 0.5) \* steps;

sum = sum + 4.0 / (1 + x \* x);

}

pi = steps \* sum;

}

public double getSum()

{

return sum;

}

public double getPi()

{

return pi;

}

}

class Program

{

static long num\_Steps = 10000000;

static double steps, final\_pi = 0.0;

static double[] values = new double[10];

static ThreadPi[] threadObjs = new ThreadPi[10];

static Thread[] threads = new Thread[10];

static void Main(string[] args)

{

Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();

int i = 0;

double x, pi, sum = 0.0;

steps = 1 / (double)num\_Steps;

for (i=0;i<threadObjs.Length;i++)

{

threadObjs[i] = new ThreadPi((Convert.ToInt32(num\_Steps / 10) \* i), Convert.ToInt32((num\_Steps / 10) \* (i + 1)), steps);

threads[i] = new Thread(new ThreadStart(threadObjs[i].ThreadFunction));

threads[i].Start();

threads[i].Join();

}

for (i = 0; i < threadObjs.Length; i++)

{

final\_pi += threadObjs[i].getPi();

}

Console.WriteLine("Final Pi = " + final\_pi);

stopWatch.Stop();

TimeSpan ts = stopWatch.Elapsed;

string elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",

ts.Hours, ts.Minutes, ts.Seconds,

ts.Milliseconds / 10);

Console.WriteLine("RunTime " + elapsedTime);

Console.Write("Enter a key to end");

Console.ReadKey();

}

}

}

**Output:**

